Criptografía y Seguridad Computacional **2016-01**

Clase 2: 09/03/2016

Profesor: Fernando Krell *Notas: Alejandro Pimentel*

1. Mundo Computacional

Dada las limitaciones que tienen los cifrados perfectamente secretos vistos en la clase anterior, introduciremos hoy una nueva definición de seguridad para esquemas de cifrado. Esta nueva definición relaja el requirimiento de que los cifrados sean *perfectamente* indistinguibles a que sean indistinguibles frente a un algoritmo efficiente (es decir, que corre en tiempo polinomial). Entonces, asumiremos que el adversario tiene un poder limitado computacionalmente. Para formalizar esta intuición, introduciremos un parametro que llamamos parámetro de seguridad y que en esta clase notaremos con λ. Intuitivamente, el parametro λ nos permite definir que tan seguro es nuestro esquema. Dejaremos que todos los algoritmos (incluido el adversario) corran en tiempo polinomial en λ, pero requerimos que la probabilidad de que el adversario gane sea negligible en λ.

Sea $\Pi = \langle \mathsf{Gen}(\cdot), \mathsf{Enc}(\cdot), \mathsf{Dec}(\cdot) \langle$ un esquema de cifrado, es decir:

- $\mathsf{Gen}(1^\lambda)$: Algoritmo aleatorio que genera una llave k. (por el momento, pensemos que k es uniformemente aleatorio en $\{0,1\}^\lambda$
- $\mathsf{Enc}(k, m)$: Algoritmo posiblemente aleatorio que dado una llave k y un mensaje m (de tamaño polinomial en λ) genera un texto cifrado c.
- $\mathsf{Dec}(k, c)$: Algoritmo determinista que descifra c (si $c \in \mathsf{Enc}(k, m)$ entonces retorna m).

Todos los algoritmos corren en tiempo polinomial en λ.

Intuitivamente queremos lo siguiente: Π es (t, ε)-seguro si $\forall$ adversario que corre en tiempo t, no puede quebrar Π con probabilidad $> \varepsilon$.

Ejemplo 1. *Sea λ 128. Estrategias del adversario:*

- *Probar todas las llaves: $|k| = 2^\lambda$, gana con $\Pr = 1$. Tiempo es exponencial en λ*
- *Probar una llave (aleatoria): tiempo es constante $O(1)$, gana con $\Pr = 2^{-128}$.*

$\underline{\mathsf{Exp}^{\mathtt{ind}}_{\Pi,\mathcal{A}}}$:

1. $k \leftarrow \mathsf{Gen}(1^\lambda)$
2. $m_0, m_1 \leftarrow \mathcal{A}(1^\lambda)$
3. $b \xleftarrow{\$} \{0,1\}$
4. $b' \leftarrow A(\mathsf{Enc}(k, m_b))$

output "$\mathcal{A}$ gana" sii $b' = b$

Figura 1: Experimento $\mathsf{Exp}^{\mathtt{ind}}_{\Pi,\mathcal{A}}$

- *Probar un numero polinomial de llaves. Gana con* $\Pr = \frac{\mathrm{poly}(\lambda)}{2^\lambda} = \frac{\mathrm{poly}(\lambda)}{2^{|k|}} = \mathsf{negl}(\lambda)$

2. Seguridad

Definimos seguridad mediante el experimento en figura 1

Definición 2. Π *tiene* cifrados computacionalmente indistinguibles *si* $\forall$ *PPT*[1] $\mathcal{A}$ $\forall\lambda$ *suficientemente grande:*

$$\Pr[\mathsf{Exp}^{ind}_{\Pi,A}(A) = \text{“}A \text{ gana”}] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

Alternativamente, podemos dar el bit b como entrada al experimento $\mathsf{Exp}^{\mathtt{ind}}_{\Pi,A}(\lambda, b)$ y definir seguridad como:

$$|\Pr[\mathsf{Exp}^{\mathtt{ind}}_{\Pi,A}(\lambda, 0) = 0] - \Pr[\mathsf{Exp}^{\mathtt{ind}}_{\Pi,A}(\lambda, 1) = 0]| \leq \mathsf{negl}(\lambda)$$

Para todo adversario PPT $\mathcal{A}$ y para todo λ suficientemente grande.

2.1. Implicancias 1

Demostraremos ahora que si un esquema Π tiene cifrados computacionalmente indistinguibles, entonces ningún adversario puede adivinar cualquier bit de un mensaje uniformemente aleatorio con probabilidad significativamente mayor a $1/2$.

[1] Probabilista de tiempo polinomial

Teorema 3. *Si* Π *tiene cifrados indistinguibles, entonces para todo adversario* $\mathcal{A}$ *y para todo* λ *suficientemente grande*

$$\Pr[\mathcal{A}(\mathsf{Enc}_k(m)) = m^i] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

para todo i*, en done la probilidad esta tomada sobre la elección aleatoria de la llave* k *y la elección uniformemente aleatoria de el mensaje* m*.*

Demostración. Supongamos que $\exists\mathcal{A}$ tal que para algún i $\Pr[\mathcal{A}(\mathsf{Enc}_k(m)) = m^i] \geq \frac{1}{2} + \frac{1}{\text{poly}(\lambda)}$.
Construiremos un adversario $\mathcal{A}'$ tal que $\Pr[\mathsf{Exp}^{\text{ind}}_{\Pi,A'}(\lambda) = \text{“A gana”}] \geq \frac{1}{2} + \frac{1}{\text{poly}(\lambda)}$.

Sea I^i_b la distribución sobre $\{0,1\}^n$ tal que todos los bits son elegidos uniformente aleatorios exceptio el i-ésimo el cual es fijado en b.

El algoritmo $\mathcal{A}'$ elige m_0 acorde a I^i_0 e m_1 acorde a I^i_1. Luego recibe el cifrado de m_b y llama a $\mathcal{A}$ para que adivine el i-ésimo bit. Si $\mathcal{A}$ retorna 0 entonces el cifrado corresponde a m_0. Si $\mathcal{A}$ responde 1, entonces el texto cifrado corresponde a m_1.

1. $A' : m_0 \sim I_0; m_1 \sim I_1$
2. $A'(enc_k(m_b))$. Ejecutamos $A \to m^i$
   - Si $m^i = 1$: output 1
   - Si $m^i = 0$: output 0

Análisis:

$$\begin{aligned}\Pr[\mathsf{Exp}^{ind}_{\Pi',A}(\lambda) = \text{“A gana”}] &= \Pr[b = b'] \\ &= \Pr[m^i = b] \\ &= \underbrace{\Pr[A(\mathsf{Enc}_k(m_b)) = b]}_{\frac{1}{2}+\frac{1}{\text{poly}(\lambda)}}\end{aligned}$$

Llegamos a una contradicción, porque $\mathcal{A}'$ gana con $\Pr \geq \frac{1}{2} + \text{poly}(\lambda)$ □

2.2. Implicacia 2

Si Π tiene cifrados indistinguibles, un adversario no puede obtener ninguna función sobre el mensaje: Si $A(c = enc_k(m)) = f(m), \exists \mathcal{B}$ tq. $\mathcal{B}(1^\lambda)$[2] $= f(m)$
Si el adversario puede obtener a partir del texto cifrado alguna función del mensaje, entonces f(m) se puede obtener sin conocer el mensaje.

$$|\Pr[A(enc_k(m)) = f(m)] = \Pr[\mathcal{B}(|m|) = f(m)]| = \mathsf{negl}(\lambda)$$

Demostración. (Idea)
$enc_k(m) \cong enc_k(1^\lambda)$
Si $\mathcal{A}$ puede obtener $f(m)$ a partir de $enc_k(m)$, también la puede obtener de $enc_k(1^\lambda)$.
Si no, usaremos $\mathcal{A}$[3] para distinguir entre $enc_k(m_0)$ y $enc_k(1^\lambda)$.

- A' que gana en $\mathsf{Exp}^{ind}_{\Pi,A'}$
- $A' : m_0 \leftarrow \$\{0,1\}^n; m_1 = 1^n$
- $a'(c = enc_k(m_b))$
- Ejecuto A(c) $\rightarrow$ g
  Si $g = f(m_0)$, entonces output 0; sino output 1.

Análisis:

$$\begin{aligned}
&\Pr[\mathsf{Exp}^{ind}_{\Pi,A'}(\lambda) = \text{“A gana”}] \\
&= \Pr[b' = b] = \Pr[b' = b \wedge b = 1] + \Pr[b' = b \wedge b = 0] \\
&= \underbrace{\underbrace{\frac{1}{2}}_{\frac{1}{2}-\Pr[b'=0|b=1]} \Pr[b' = b|b = 1]}_{\Pr[A(enc_k(1^n))=f(m)]} + \underbrace{\underbrace{\frac{1}{2}}_{\Pr[A(m)=f(m)]} \Pr[b' = b|b = 0]}_{\Pr[A(enc_k(1^n))=f(m)]} \\
&= \frac{1}{2} + \frac{1}{2}\Pr[A(enc_k(m)) = f(m)] - \frac{1}{2}\Pr[A(enc_k(1^\lambda)) = f(m)] \\
&\geq \frac{1}{2} + \frac{1}{\text{poly}(\lambda)}
\end{aligned}$$

□

[2] $1^\lambda = |m|$
[3] $\exists \mathcal{A}$ tq. $|\Pr[A(enc_k(m)) = f(m)] - \Pr[A(enc_k(1^n)) = f(m)]| \geq \frac{1}{\text{poly}(\lambda)}$

2.3. Seguridad Semántica

Π es *semánticamente seguro*, si $\forall PPT\mathcal{A}\ \exists PPT\mathcal{A}'$ tal que $\forall m \forall f \forall h\ \exists \mathsf{negl}(\lambda)$:

$$|\Pr[\mathcal{A}(1^\lambda, \mathsf{Enc}_k(m), h(m)) = f(m)] - \Pr[\mathcal{A}'(1^\lambda, h(m)) = f(m)]| \leq \mathsf{negl}(\lambda)$$

En palabras simples, significa que cualquier información que se puede obtener a partir del texto cifrado, no requiere del texto cifrado.

Teorema 4. *Seguridad semántica* $\equiv$ *cifrados indistinguibles*

3. Pseudoaleatoriedad

Sea $\mathcal{X}_n$ una distribución sobre $\{0,1\}^n$. Decimos que $\mathcal{X}$ es una distribución pseudo-aleatoria si nungún algoritmos puede distringuir entre strings elegidos adorde a X_n y strings elegidos acorde a la distribución uniforme U_n. En otras palabras, un string es pseudoaleatorio si se ve aleatorio para todo algoritmo de tiempo polinomial.

Observación 5. *Si $c = enc_k(m)$ es pseudoaleatorio, entonces tenemos cifrados computacionalmente indistinguibles. Idea: Podemos usar el one-time pad en donde la llave la obtenemos de una distribución pseudo-aleatoria X_n.*

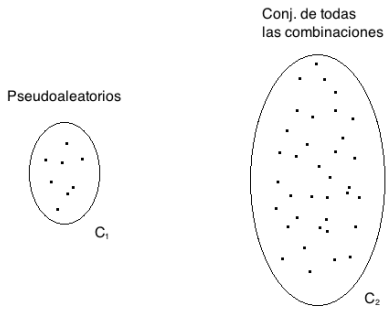
Lo que necesitamos es una función G que dado un string de largo λ (la llave), genere un string de largo n (largo de los mensages) y que este string sea pseudo-aleatorio.

Definición 6 (Generadores Pseudo-aleatorios (PRF))**.** *Una función G es un generador pseudoaleatorio de expansión $l(\cdot)$ si:*

- *Expande. $G : \{0,1\}^\lambda \to \{0,1\}^{\ell(\lambda)}$ para un polinomio $\ell(\lambda) > \lambda$*
- *Efficientemente computable. Existe un algoritmo de tiempo polinomial talque dado x genere $G(x)$.*
- *Pseudo-aleatorio. Para todo algoritmo PPT distinguidor $\mathcal{D}$, existe una función negligible* negl *tal que*

$$|\Pr_{x\sim U_\lambda}[\mathcal{D}(1^\lambda, G(x)) = 1] - \Pr_{u\sim U_{l(\lambda)}}[\mathcal{D}(1^\lambda, u) = 1]| \leq \mathsf{negl}$$

El conjunto generado por G es un subconjunto pequeño de $\{0,1\}^\ell\lambda$ (tiene solo 2^λ elementos). Es decir, estadisticamente la distribución inducida por G, aun cuando su input es uniformemente aleatorio, es extremadamente diferente a U_n.



Sin embargo, la pseudo-aleatoriedad de G nos asegura que cualquier algoritmo pude usar $G(x)$ como si fuera $100\,\%$ aleatorio sin perder seguridad de nuestro sistemas.